This Page is Inserted by IFW Indexing and Scanning Operations and is not part of the Official Record

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ BLACK BORDERS
- ☐ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- ☑ FADED TEXT OR DRAWING
- ☐ BLURRED OR ILLEGIBLE TEXT OR DRAWING
- ☐ SKEWED/SLANTED IMAGES
- ☐ COLOR OR BLACK AND WHITE PHOTOGRAPHS
- ☐ GRAY SCALE DOCUMENTS
- ☐ LINES OR MARKS ON ORIGINAL DOCUMENT
- ☐ REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- ☐ OTHER: _____

IMAGES ARE BEST AVAILABLE COPY.
As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.

This Page Blank (uspto)

PCT/EP 0 0 / 0 3 5 3 0

BUNDESREPUBLIK DEUTSCHLAND

EP00/03530

REC'D 30 MAY 2000
EPO - Munich
WIPO PCT

03. Mai 2000

PRIORITY DOCUMENT
SUBMITTED OR TRANSMITTED IN COMPLIANCE WITH RULE 17.1(a) OR (b)

4

Bescheinigung

09/926376

Die Giesecke & Devrient GmbH in München/Deutschland hat eine Patentanmeldung unter der Bezeichnung

"Sicherung eines Rechnerkerns gegen äußere Manipulationen"

am 23. April 1999 beim Deutschen Patent- und Markenamt eingereicht.

Die angehefteten Stücke sind eine richtige und genaue Wiedergabe der ursprünglichen Unterlagen dieser Patentanmeldung.

Die Anmeldung hat im Deutschen Patent- und Markenamt vorläufig das Symbol G 06 F 12/14 der Internationalen Patentklassifikation erhalten.

München, den 25. April 2000

Deutsches Patent- und Markenamt

Der Präsident

Im Auftrag

Dzierzon

Aktenzeichen: 199 18 620.0

A 9161
06.90
11/98

Sicherung eines Rechnerkerns gegen äußere Manipulationen

Die vorliegende Erfindung betrifft die Sicherung eines Rechners gegen äußere Manipulationen, insbesondere die Sicherung der im Rechnerkern bzw. zentralen Verarbeitungseinheit (CPU) vorliegenden Daten. Diese Erfindung findet insbesondere Anwendung bei Chipkarten, da diese gegen Manipulationen von außen besonders gesichert sein müssen.

Es ist bekannt, Speicherbereiche eines Rechners zum Beispiel durch Busverschlüsselung, Speicherverschlüsselung und dergleichen gegen Manipulationen zu schützen. Aus DE 37 09 524 C2 ist beispielsweise eine Prüfroutine zur Überprüfung der Speicherzelleninhalte eines Programmspeichers bekannt. Durch Prüfsummenbildung über die Speicherzelleninhalte jeweils zu Beginn oder während eines laufenden Programms und Vergleich mit einer im Programmspeicher zuvor abgelegten Prüfsumme läßt sich eine Veränderung der ursprünglichen Speicherzelleninhalte oder auch eine erst im Betrieb auftretende Veränderung erkennen, was zu einer Fehlermeldung führt.

Aufgabe der vorliegenden Erfindung ist es, eine Möglichkeit vorzuschlagen, wie der Rechner gegen äußere Manipulationen besser gesichert werden kann.

Die Aufgabe wird erfindungsgemäß durch ein Verfahren, durch eine zentrale Verarbeitungseinheit zur Durchführung dieses Verfahrens und durch einen Rechner bzw. eine Chipkarte mit einer solchen zentralen Verarbeitungseinheit gemäß den Merkmalen der nebengeordneten Ansprüche gelöst. In den Unteransprüchen sind vorteilhafte Ausgestaltungen der Erfindung angegeben.

Die Erfindung geht davon aus, daß durch eine Sicherung der im Rechnerkern, das heißt in der zentralen Verarbeitungseinheit (CPU) des Rechners, vorliegenden Daten gegen äußere Manipulationen die Sicherheit des Rechners erhöht werden kann, da im Rechnerkern die Daten unverschlüsselt vorliegen und daher leicht manipulierbar sind.

Um derartige Manipulation zu erkennen, wird nach der Verarbeitung eines Befehls durch die CPU aus mehreren Registerinhalten der CPU durch mathematische Verknüpfung, beispielsweise durch Exklusiv-Oder-Verknüpfung (XOR-Verknüpfung), eine Checksumme ermittelt und als End-Checksumme in einem Speicher abgelegt. Vor der Verarbeitung des nächsten Befehls durch die CPU wird erneut eine Checksumme gebildet, das ist die Anfangs-Checksumme. Durch Vergleich der Anfangs-Checksumme mit der End-Checksumme, die übereinstimmen müssen, läßt sich feststellen, ob Registerinhalte der CPU nach der letzten Befehlsverarbeitung manipuliert wurden. Als Registerinhalte kommen die Inhalte jener Bereiche der CPU in Betracht, die einen von Null verschiedenen Zustand annehmen können, wie z.B. bei dem Prozessortyp 8051 der Accu, B-Accu, Datapointer (DPTR, DPL, DPH), Register (R0 bis R7) der Registerbänke, Programmstatuswort (PSW), Stackpointer (SP), Special Function-Register (SPR) und dergleichen.

Zur weiteren Erhöhung der Sicherheit kann zusätzlich beim Laden eines Befehls ein Zähler gestartet werden, welcher die Taktzyklen, die zur Abarbeitung des Befehls notwendig sind, zählt. Der Zähler ist dabei vorzugsweise hardwaremäßig aufgebaut. Eine Logik entnimmt dem Befehlsopcode die Anzahl der zur Abarbeitung notwendigen Taktzyklen und setzt diese in einen Zählerwert um. Der Zähler läuft dann parallel zum ausgeführten Befehl.

Es wird überprüft, ob der abzuarbeitende Befehl innerhalb der angegebenen Taktzyklen abgearbeitet wird. Für den Fall, daß der Befehl nicht innerhalb des vorgegebenen Zeitraums abgearbeitet wurde, wird beispielsweise die Taktversorgung eingestellt, so daß eine weitere Abarbeitung von Befehlen nicht mehr möglich ist. Alternativ kann auch ein Reset ausgelöst und somit die Zentraleinheit zurückgesetzt werden. Die gleichen Maßnahmen können getroffen werden, wenn der Befehl vorzeitig abgearbeitet wurde, d.h. wenn dem Befehlszähler noch nicht in seinem Grenzwert angelangt ist und bereits ein neuer Operationscode erkannt wurde.

Die logische Verknüpfung der sicherheitsrelevanten Register kann durch Hardware oder Software realisiert werden. Die Checksummenbildung zwischen zwei aufeinanderfolgenden Befehlen kann beispielsweise aufgrund zufälliger oder definierter Ereignisse oder ständig erfolgen.

Die Erfindung wird nachfolgend anhand der Zeichnungen näher erläutert. Es zeigen:

Fig. 1 den Aufbau eines Mikrocontrollers am Beispiel eines 8051-Prozessors und

Fig. 2 eine Logik zur Verknüpfung mehrerer Bereiche der zentralen Verarbeitungseinheit.

Fig. 1 zeigt den Aufbau eines 8051-Prozessors, das ist ein 8-Bit-Prozessor. Während mit bekannten Verschlüsselungsverfahren die Daten durch Bus- oder Speicherverschlüsselung vor Manipulation geschützt werden, liegen die Daten im Kern des Rechners, d. h. in der zentralen Verarbeitungseinheit bzw. CPU, unverschlüsselt vor. Mit dem erfindungsgemäßen Verfahren

wird nun ermittelt, ob eines oder mehrere Register der CPU manipuliert worden sind.

In Fig. 2 sind beispielhaft solche sicherheitsrelevanten Bereiche der CPU angegeben, die manipuliert werden könnten, nämlich Stackpointer SP, Akku AC, B-Akku BAC, Register RO bis R7, Data-Pointer DPL und DPH zu den unteren bzw. oberen Bereichen des internen RAM. Diese Register werden miteinander logisch verknüpft, um eine Checksumme zu bilden. In Fig. 2 sind jeweils zwei 8-Bit-Register durch ein Exklusiv-Oder-Gatter (XOR) miteinander verknüpft. So ergibt sich aus der XOR-Verknüpfung der Register RO und ein neues 8-Bit-Muster, das wiederum XOR-verknüpft wird mit dem 8-Bit-Muster, das sich aus der XOR-Verknüpfung der Register R1 und R7 ergibt. Durch weitere XOR-Verknüpfung der sich jeweils ergebenden 8-Bit-Muster ergibt sich schließlich ein 8-Bit-Muster, das als Checksumme dient und in Fig. 2 mit „Anfangs-Checksumme" bezeichnet ist. Anstelle der XOR-Verknüpfung, welche insbesondere hinsichtlich des Aufwandes vorteilhaft ist, können selbstverständlich auch andere Ausführungsformen zur Bildung der Checksumme gewählt werden.

Wenn die Verknüpfung der Register hardwaremäßig durch logische Verknüpfungsglieder ausgeführt ist, ändert sich die Checksumme unmittelbar, wenn sich der Inhalt eines Registers ändert. D. h. , während der Abarbeitung eines in der CPU verarbeiteten Befehls ändert sich die Checksumme gegebenenfalls mehrfach. Entscheidend für die Durchführung des Verfahrens sind aber nur die Checksumme nach Abarbeitung eines Befehls und vor Abarbeitung des nächstfolgenden Befehls, da diese beiden Checksummen (End-Checksumme des einen Befehls und Anfangs-Checksumme des nächstfolgenden Befehls) in einem Komparator miteinander verglichen werden.

Der Vergleich wird wie folgt durchgeführt: Die sich am Ende der Abarbeitung eines ersten Befehls einstellende Checksumme wird als End-Checksumme in einem Speicher auf der CPU abgelegt. Um nun festzustellen, ob nach der Abarbeitung dieses ersten Befehls und vor dem Laden des nächstfolgenden, zweiten Befehls in die CPU eine Manipulation der CPU erfolgt ist, wird parallel zum Laden dieses zweiten Befehls die Anfangs-Checksumme wie eingangs beschrieben gebildet. In einem ersten Schritt a.) wird die Anfangs-Checksumme mit der in dem Speicher abgelegten End-Checksumme aus dem zuvor abgearbeiteten ersten Befehl mittels eines Komparators verglichen. Für den Fall, daß keine Manipulation an der CPU vorgenommen worden ist, stimmen Anfangs- und End-Checksumme überein und der Wert des Vergleichsergebnisses ist Null. Der Komparator gibt ein Signal aus, aufgrund dessen in einem zweiten Schritt b.) nach Abarbeitung des zweiten Befehls die gerade anliegende Checksumme als neue End-Checksumme in den Speicher eingespeichert wird. D. h. , die Abarbeitung des zweiten Befehls wird in diesem Falle nicht unterbrochen. Ergibt sich hingegen beim Vergleich der Anfangs-Checksumme mit der End-Checksumme ein Wert ungleich Null, so ist auf eine Manipulation der CPU zu schließen. Das Ausgangssignal des Komparators veranlaßt dann anstelle des zweiten Schritts b.) eine Fehlermeldung c.), die in dem in Fig. 2 dargestellten Fall einen Abbruch der Befehlsverarbeitung verursacht. Beispielsweise kann der Prozessor angehalten werden, ein Sicherheitssensor kann aktiv geschaltet werden oder im Falle einer Chipkarte kann die Chipkarte vom Terminal einbehalten werden.

Der zuvor beschriebene Sicherheitsmechanismus kann auch rein softwaremäßig realisiert werden, indem die Checksummen einerseits am Ende einer Befehlsabarbeitung und andererseits zu Beginn der nächsten Befehlsabarbeitung ermittelt und miteinander verglichen werden. Das entsprechende Pro-

gramm kann beispielsweise in dem ROM bzw. EPROM des Prozessors gespeichert sein und die End-Checksumme kann in dem bitadressierbaren RAM des Prozessors abgelegt werden.

Das beschriebene Verfahren braucht nicht vor jedem abzuarbeitenden Befehl durchgeführt zu werden. Eine Ausgestaltung der Erfindung sieht vor, daß die Durchführung des Verfahrens von einem zufälligen oder einem definierten Ereignis abhängt. Gemäß einer ersten Ausführungsform kann das Verfahren zeitabhängig getriggert werden.

Gemäß einer anderen Ausführungsform kann das Verfahren dadurch getriggert werden, daß der Inhalt eines oder mehrerer Register der CPU einem vorbestimmten Muster entspricht.

Eine noch weitere Ausführungsform der Erfindung sieht vor, daß das Verfahren jeweils nach Verarbeitung einer vorgegebenen Anzahl von Befehlen getriggert wird.

Bevorzugt wird eine Ausführungsform, wonach das Verfahren nur dann getriggert wird, wenn zwischen dem Befehl, nach dessen Abarbeitung die Checksumme als End-Checksumme in den Speicher gespeichert wurde, und der Anfangs-Checksumme zu Beginn der Abarbeitung des nächstfolgenden Befehls eine längere, definierte Zeitspanne liegt. Dadurch wird wertvolle Rechnerkapazität bei der Ausführung eines Programms mit vielen Befehlen gespart. Wenn man davon ausgeht, daß eine Manipulation der CPU, insbesondere bei Chipkarten, nicht während des laufenden Programms stattfindet, sondern wenn die Chipkarte aus dem Chipkartenterminal entfernt ist, so ist eine Manipulation der CPU mittels dieser zuletzt beschriebenen Ausführungsform dennoch zuverlässig feststellbar.

Patentansprüche

1. Verfahren zur Sicherung eines Rechners mit zentraler Verarbeitungseinheit (CPU) gegen äußere Manipulation, dadurch **gekennzeichnet**, daß anhand von sich am Ende der Verarbeitung eines Befehls durch die CPU einstellenden Registerinhalten der CPU durch mathematische Verknüpfung eine End-Checksumme gebildet und gespeichert wird und anhand der sich vor Beginn der Verarbeitung des nächstfolgenden Befehls durch die CPU einstellenden Registerinhalte eine Anfangs-Checksumme gebildet wird, wobei eine Fehlermeldung erfolgt, wenn die Anfangs-Checksumme nicht mit der End-Checksumme übereinstimmt.

2. Verfahren nach Anspruch 1, **dadurch gekennzeichnet, daß** beim Laden des Befehls ein Zähler gestartet wird, welcher die Taktzyklen, die zur Abarbeitung des Befehls notwendig sind, zählt und bei Über- oder Unterschreiten der vorgegebenen Taktzyklen ein Fehlersignal ausgibt.

3. Verfahren nach Anspruch 2, **dadurch gekennzeichnet, daß** das Fehlersignal einen Interrupt auslöst oder zur Einstellung der Taktsignalversorgung führt.

4. Verfahren nach einem der Ansprüche 1 bis 3, **dadurch gekennzeichnet, daß** die Anzahl der zur Abarbeitung eines Befehls notwendigen Taktzyklen aus dem OP-Code des Befehls durch eine Logikschaltung erhalten wird.

5. Verfahren nach einem der Ansprüche 1 bis 4, dadurch **gekennzeichnet**, daß die mathematische Verknüpfung mittels Exklusiv-Oder-Verknüpfung der Registerinhalte erfolgt.

6. Verfahren nach einem der Ansprüche 1 bis 5, dadurch **gekennzeichnet,** daß die Initiierung des Verfahrens durch zufällige oder definierte Ereignisse getriggert wird.

7. Verfahren nach Anspruch 6, dadurch **gekennzeichnet,** daß das Verfahren zeitabhängig getriggert wird.

8. Verfahren nach Anspruch 6, dadurch **gekennzeichnet,** daß das Verfahren getriggert wird, wenn der Inhalt eines oder mehrerer Register der CPU einem vorbestimmten Muster entspricht.

9. Verfahren nach Anspruch 6, dadurch **gekennzeichnet,** daß das Verfahren jeweils nach Verarbeitung einer vorgegebenen Anzahl von Befehlen getriggert wird.

10. Zentrale Verarbeitungseinheit (CPU) für einen Rechner zur Durchführung des Verfahrens nach einem der Ansprüche 1 bis 6, umfassend

- eine Verknüpfung mehrerer Register der CPU durch logische Verknüpfungsglieder zur Bildung einer Checksumme,
- einen Checksummenspeicher zur Speicherung einer von den logischen Verknüpfungsgliedern gebildeten, ersten Checksumme,
- einen Komparator zum Vergleichen einer von den logischen Verknüpfungsgliedern gebildeten, zweiten Checksumme mit der im Speicher gespeicherten, ersten Checksumme und
- eine Steuereinrichtung zum Steuern der Einspeicherung der ersten Checksumme in den Checksummenspeicher und zum Steuern des Komparators.

11. Zentrale Verarbeitungseinheit nach Anspruch 10, **gekennzeichnet durch** einen Zähler zum Zählen der für eine Befehlsabarbeitung benötigten Taktzyklen.

12. Zentrale Verarbeitungseinheit nach Anspruch 10 oder 11, **gekennzeichnet durch** eine Logikschaltung zum Bestimmen der für die Abarbeitung eines Befehls notwendigen Taktzyklen aus dem OP-Code des Befehls.

13. Rechner umfassend eine zentrale Verarbeitungseinheit nach einem der Ansprüche 10 bis 12.

14. Chipkarte umfassend eine zentrale Verarbeitungseinheit nach einem der Ansprüche 10 bis 12.

Zusammenfassung

Es wird ein Verfahren zur Sicherung der zentralen Verarbeitungseinheit eines Rechners, insbesondere einer Chipkarte, vorgeschlagen. Einzelne sicherheitsrelevante Register werden logisch miteinander verknüpft um eine Checksumme zu bilden, nachdem die CPU einen Befehl abgearbeitet hat. Diese Checksumme wird gespeichert und vor Beginn der Verarbeitung des nächstfolgenden Befehls mit einer entsprechend gebildeten Checksumme verglichen. Stimmen die miteinander verglichenen Checksummen nicht überein, so deutet das auf eine Manipulation der Registerinhalte der CPU im Zeitraum zwischen der Abarbeitung der beiden Befehle hin. In einem solchen Fall ergeht eine entsprechende Fehlermeldung und der Prozessor wird angehalten, oder die Karte wird eingezogen.

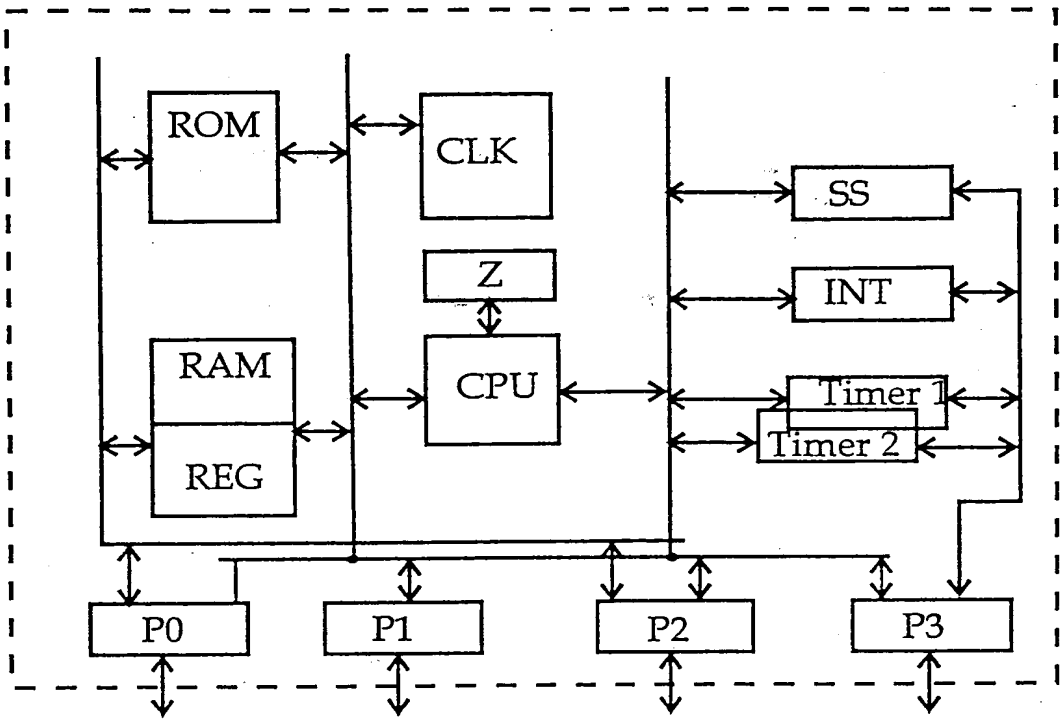
ROM
CLK
SS
Z
INT
RAM
CPU
Timer 1
REG
Timer 2
P0
P1
P2
P3

Fig. 1

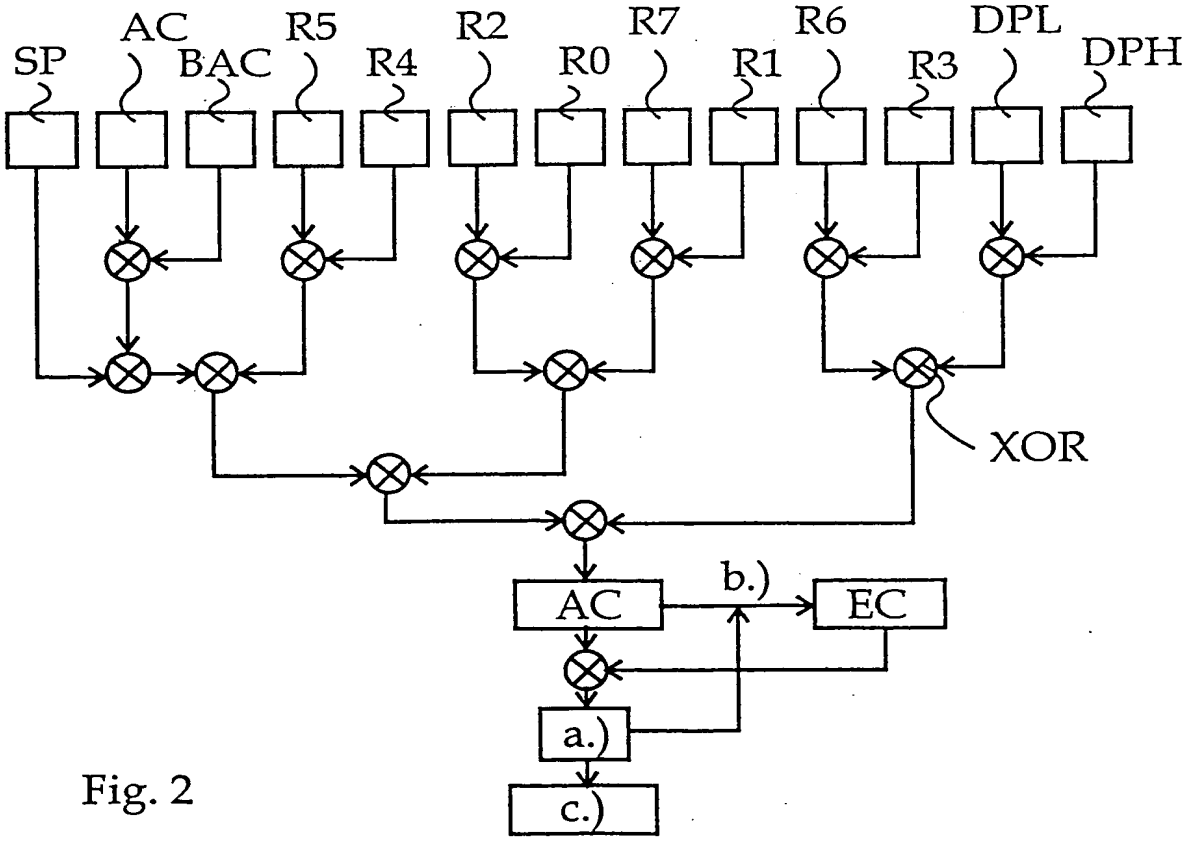
SP
AC
BAC
R5
R4
R2
R0
R7
R1
R6
R3
DPL
DPH
XOR
AC
b.)
EC
a.)
c.)

Fig. 2